



National Certificate of Educational Achievement
TAUMATA MĀTAURANGA Ā-MOTU KUA TAEA

Internal Assessment Resource 2014

Digital Technologies Level 1

This resource supports assessment against:

Achievement Standard 91076 v2

Construct a basic computer program for a specified task

Resource title: Create a Game

3 credits

Student Name:

.....

I declare that the material I have submitted for this unit or achievement standard is my own work and that I had no outside help from others in completing it.

(student to sign)

Comments:

.....
.....
.....
.....

☐ Not Achieved ☐ Achieved ☐ Merit ☐
Excellence



DEPARTMENT USE ONLY

Internal Moderation Grade: _____ Signed: _____ Date: _____

If your grade differs from the mark given by the teacher, fill in the 'internal moderation' report.

Student Instructions

Introduction

This assessment activity requires you to construct a basic computer program that is educational. You will be assessed on how skilfully you construct the program and the efficiency of the program.

This is an individual task. You have 5 weeks (25 hours) to complete this task. It needs to be submitted by Friday the 23rd of May. Due to the nature of the task, there will be no reassessment opportunity for this standard.

Task

Create:

Using the plan you created for AS91075, or a plan provided for you, construct a computer game. This will either be a racing game, or another game of your choice that has been discussed with a teacher.

Specifications:

Your computer program must include the following:

- At least two variables *e.g. speed, driver_name* storing at least two types of information *e.g. number and text*
- Accurate and explanatory variable names
- Assignment of variables *e.g. set score to 0*
- Predefined actions *e.g. switch to costume, say "You win!"*
- User input *e.g. keyboard, mouse*
- Use of a sequence control structure *i.e. several statements run in a row*
- Use of a selection control structure *e.g. if, if-else*
- Use of an iterative control structure *e.g. repeat until, forever.*
- Comments which describe what each script does
- Comments to explain any complicated pieces of code

Your submission must also include:

- A testing log demonstrating that comprehensive testing and debugging occurred to identify and fix errors in the program's functionality.

Test and debug:

Using your test plan, which you can add to if needed, make sure you test your program thoroughly and try to fix any bugs you find. Create a testing log to provide evidence of this testing and debugging. Create this log as you work on your program. Refer to Appendix 1. Consider:



- Does the program behave as it should with **expected inputs** *e.g. that a number was entered if the user was asked to enter a number.*
- Does the program behave as it should when boundary inputs have been reached?
- How does the program behave if it receives **invalid inputs** *e.g. the user entered text when they were asked for a number.*

Prior to submitting your work, refer to *Appendix 2 - Assessor Checklist* and verify you are able to tick all the boxes for the Achieved, Merit or Excellence. The assessor will tick off these items as they have been observed.

Submit:

Hand a copy of the following in to your teacher:

1. Print out of the source code for your program.
2. Electronic copy of your program.
3. Print out of your Testing/Development Log

Appendix 1 - *Partial Testing Log - Using example test plan for Asteroids game*

7/5/14

Beginning of game: Score & lives are set correctly, Forgot to ask for name. Fixed that and retested it.

8/5/14

Start of game all ok. Tested game play - turning the ship left & right works. Space bar does shoot, but the bullet is going the wrong way. Will fix tomorrow.

9/5/14

During game. Ship controls (left/right) fine. Shooting in right direction now. Rock explodes when hit. So far only one rock. It does reappear after exploding, but always at the same point on the edge of the screen. Rock continuously moves towards ship.

Etc...

Appendix 2 - Assessor Checklist:

Requirements	A	M	E
The student has constructed a basic computer program which addresses the specified task by implementing the program plan.	<input type="radio"/>		
The student constructed the program independently and accurately <i>e.g. you take responsibility for creating, documenting, testing and debugging. You make all the decisions. You construct the program on your own without advice from anyone. There are no unexpected errors. Troubleshoot problems promptly.</i>		<input type="radio"/>	
The program has been coded in a suitable programming language.	<input type="radio"/>		
The program includes:			
• Input from a user	<input type="radio"/>		
• Assignment of variables	<input type="radio"/>		
The program code is documented with:			
• Comments	<input type="radio"/>		
• Variable names and comments that accurately describe the function and behaviour of the code		<input type="radio"/>	
• Variable names and brief comments that explain and justify coding decisions			<input type="radio"/>
The program specifies variable (incl. at least two data types).	<input type="radio"/>		
Constants, variables and/or derived values are used, rather than literals.			<input type="radio"/>
The program code is clearly set out and includes procedural structure that combines: <ul style="list-style-type: none"> • Actions (individual steps) • Conditions (logical expressions within control structures) • Sequence control structures • Selection control structures • Iteration control structures 	<input type="radio"/>		
The actions, conditions and control structures are well-chosen (i.e. they ensure the task is correctly performed with no unintended behaviour or consequences)		<input type="radio"/>	
The program includes:			
• Validity checks for input data			<input type="radio"/>
• Ways for correctly handling expected, boundary and invalid inputs			<input type="radio"/>
• Individual control structures which each have a clear and well defined purpose within the structure of the plan			<input type="radio"/>
The program code is concise and there is no unnecessary duplication/repetition of control structures			<input type="radio"/>
The student has tested and debugged the program to ensure it works on the following inputs:			
• Sample and expected inputs	<input type="radio"/>		
• Expected and boundary		<input type="radio"/>	
• Expected, boundary and invalid			<input type="radio"/>

The student tested and debugged the program in an:			
• Organised way		<input type="radio"/>	
• Organised and time effective way eg. <i>You address errors as soon as they are identified. Checking parts of the program as its created, rather than upon completion.</i>			<input type="radio"/>

Assessment schedule: Digital Technologies 91076 - An Educational Computer Program

Evidence/Judgements for Achievement	Evidence/Judgements for Achievement with Merit	Evidence/Judgements for Achievement with Excellence
<p>The student has constructed a basic computer program for a specified task.</p> <p>The student has, with guidance:</p> <ul style="list-style-type: none"> implemented a plan for a basic program in a suitable programming language. <p>The program:</p> <ul style="list-style-type: none"> obtains and uses input from a user via keyboard and/or mouse includes at least two variables types (e.g. UserName - text and Score - number) assignment e.g. set UserName to... predefined methods (e.g. functions such as pick random) sequence (e.g. steps in order) selection (e.g. if, else conditions) iteration (e.g. repeat until) <ul style="list-style-type: none"> set out the program code clearly and documented the program with comments <p>The student has included comments for each of their main scripts.</p> <p>The student has set out the scripts clearly and placed them in a logical order within the Scripts window.</p>	<p>The student has skilfully constructed a basic computer program for a specified task. The student has:</p> <ul style="list-style-type: none"> independently implemented the plan for a basic program in a suitable programming language that uses a procedural structure with well-chosen actions, conditions and control structures <p>The student independently implemented the plan for the educational game to meet specifications.</p> <p>The student's program includes:</p> <ul style="list-style-type: none"> obtains and uses input from a user via keyboard and/or mouse includes at least two variables types (e.g. UserName - text and Score - number) assignment e.g. set UserName to... predefined methods (e.g. functions such as pick random) sequence (e.g. steps in order) selection (e.g. if, else conditions) iteration (e.g. repeat until) appropriate variable names <p>The procedural structure of the student's program demonstrates well-chosen actions, conditions and control structures. For example:</p> <ul style="list-style-type: none"> the resulting sequence of actions correctly performs the task. There is no unintended behaviour or consequences. 	<p>The student has efficiently constructed a basic computer program for a specified task.</p> <p>The student has:</p> <ul style="list-style-type: none"> constructed a basic program which uses actions, conditions and control structures effectively to increase the flexibility and robustness of the program <p>This could include for example – input data validation; correctly handling expected, boundary and invalid inputs; using constants, variables and derived values rather than literals;</p> <ul style="list-style-type: none"> used an effective procedural structure that results in a well-structured, logical solution to the task <p>The procedural structure of the student's program uses control structures with a well-defined purpose and contain no unnecessary duplication or repetition.</p> <ul style="list-style-type: none"> set out the program code concisely and documented the program with succinct comments that explain and justify decisions <p>The student has set out the scripts clearly and placed them in a logical order within the Scripts window.</p> <p>The student has used accurate variable names which describe the variable's purpose within the program and has used a consistent naming convention such as underscores or camelCase</p> <p>The student has included succinct comments that explain and justify decisions</p>

<ul style="list-style-type: none"> • tested and debugged the program to ensure that it works on a sample of expected inputs. <p><i>The educational game functions correctly in normal game play.</i></p> <p><i>The student has checked that the game works correctly on expected inputs (e.g. when a number is entered if the user is asked for a number).</i></p> <p><i>This description relates to only part of what is required, and is indicative only.</i></p>	<ul style="list-style-type: none"> • documented the program with variable names and comments that accurately describe code function and behaviour <p><i>The student has set out the scripts clearly and placed them in a logical order within the Scripts window.</i></p> <p><i>The student has used accurate variable names which describe the variable's purpose within the program and has used a consistent naming convention such as underscores or camelCase (e.g. User_name or UserName).</i></p> <p><i>The student has included comments that accurately describe the function and behaviour of their Scripts.</i></p> <ul style="list-style-type: none"> • tested and debugged the program in an organised way to ensure that it works on expected and boundary inputs. <p><i>The student provided annotated screenshots of testing the program incrementally throughout the development of the educational game program (e.g. the student tested the UserName section of the script correctly assigns the user input to the variable before proceeding to the next stage of development).</i></p> <p><i>The student tested expected inputs as well as boundary cases. For example, the student tested:</i></p> <ul style="list-style-type: none"> – <i>that the user entered a number when they were asked for input.</i> – <i>that the game functions as expected when the number of correct answers equals 3 and does not change level until that is reached.</i> <p><i>This description relates to only part of what is required, and is indicative only.</i></p>	<ul style="list-style-type: none"> • comprehensively tested and debugged the program in an organised and time-effective way to ensure the program is correct on expected, boundary and invalid inputs. <p><i>The student used a testing plan to test the program incrementally throughout the development of the educational game program</i></p> <p><i>The student had players test the game throughout the development of the program (not just at the end).</i></p> <p><i>The student tested expected inputs as well as boundary cases and invalid inputs. For example, the student tested:</i></p> <ul style="list-style-type: none"> – <i>that the user entered a number when they were asked for input.</i> – <i>that the game functions as expected when the number of correct answers equals 3 and does not change level until that is reached.</i> – <i>the program still works if user enters a letter rather than a number.</i> <p><i>This description relates to only part of what is required, and is indicative only.</i></p>
--	--	---